INTRODUCTION TO CPP

PROGRAM: is defined as set of instructions

PROGRAMMING: is the art of writing programs.

PROGRAMMING METHODOLOGIES are classified into 2 types:

1. PROCEDURE ORIENTED PROGRAMMING (POP)
2. OBJECT ORIENTED PROGRAMMING (OOP)

Procedure oriented programming Vs. Object oriented programming

- In POP, importance is given to the sequence of things to be done i.e. algorithms and in OOP, importance is given to the data.
- In POP, larger programs are divided into functions and in OOP, larger programs are divided into objects.
- POP follows a top down approach in problem solving while OOP follows a bottom up approach.
- In POP, there is no access specifier and in OOP there are public, private and protected specifier.
- In POP, operator cannot be overloaded and in OOP operator can be overloaded.
- In POP, Data moves openly around the system from function to function, In OOP objects communicate with each other through member functions
- COBOL, PASCAL, C, FORTRAN are some of the examples of POP languages.
- C++, JAVA are some of the examples of OOP languages.

Advantages of POP:

- Easy to read the code since functions are used and also it is easy to debug the code.

Disadvantages of POP:

- Code reusability is not permitted.
- No security for global data.

- Since no code reusability is there, the size of program will keep on increasing. As a result at a particular point the programmer looses the control over the code that is, flow of execution of code cannot be understood.

Benefits of OOPL:

- Inheritance can eliminate redundancy of code and provide reusability.
- Data hiding helps to built secure programs
- Easy to partition a task into object
- Data central design approach enables us to capture more details of a model in implemented form.

Why we go for OOPS…?

Because this is an implementation over structure programming.

Advantages of OOPS:

- **Extendibility increases**
- **Reusability increases**

HISTORY OF C++ :

C++ is an Object Oriented Programming Language. It was developed by Bjarne Stroustrup at AT and T Bell Labs during 1980's. C++ is an extension of C with major addition of all object oriented features. Stroustrup initially called C++ as "C with Classes". C++ is a suoperset of C. therefore all the programs in C are also written in C++. He had combined the Simula's use of classes and object-oriented features with the power and efficiency of C. The term C++ was first used in 1983.

PROPERTIES OF OOPS

1) OBJECT
2) CLASS
3) ABSTRACTION
4) ENCAPSULATION
5) INHERITANCE
6) POLYMORPHISM
7) DYNAMIC BINDING
8) MESSAGE PASSING

OBJECT:

- Anything that exists in the real world is said to be object
- An object may be a name of person, place or thing etc.,,
- Every object will have some property and behavior.
- Object is a variable of type class.

CLASS:

- Class is a collection of objects
- Class is a common name given to a group of objects
- For example: IT is name of class. Each and every student present in IT class is said to be a object.
- In computer terminology class is called as collection of data and functions.
- Properties of object are represented by data (variables)
- Behaviour of object is represented by functions of the class.

Example:

1. Consider "student" as object

Properties of student: name, roll no, marks, height.weight, gender,color etc.,, Behavior of student: writing, reading, listening etc.,,

2.

objects	class
Apple, mango,grapes,orange etc.,,	fruit
Pink,blue,black,yellow etc.,,	color

DATA ABSTRACTION:

- The act of representing essential features with out including back ground details (Hiding the actual content and showing only the required content is said to be abstraction).
- Example: index of text book, google search engine etc.,,
- **Advantage of abstraction is** every user will get his/her own view of the data.

DATA ENCAPSULATION:

- Wrapping (combining) up of data and function into a single unit is called as encapsulation.
- Data will be not accessible to external classes. Only those functions that are present in that class can access that data.

INHERITANCE:

- One class using the properties of another class is said to be inheritance. Ex: parents - children
- Code reusability is achieved through inheritance
- Class whose properties are used by other class is called as **BASE CLASS.**
- Class which uses the properties of other class is said to be **DERIVED CLASS**.

PLOYMORPHISM

- Ability to take more than one form is said to be polymorphism.
- POLY means MANY
- MORPHISM means FORMS

- Example:

  I. + is used for addition and concatination
  II. ***** is used for multiplication and also for declaring a pointer variable
  III. **>> and <<** are used for right and left shift of bit operations and also along with cin and cout in CPP.

DYNAMIC BINDING

- Link between function call and function procedure is made at run- time.
- Dynamic binding is also called as late binding or run-time binding.

MESSAGE PASSING:

- In OOP, set of objects communicate with each other.

Structure of C++:

1. Include files
2. Class declaration
3. Member functions
4. Definitions
5. Main function program

COUT & CIN

- cout is used to print data on the screen.
- cin is used to accept values at run-time.

//Simple Program (Program to add two integers)

```
#include<iostream>
using namespace std;
main()
```

```
{
    int a,b,c;
    cout<<"enter 2 integers";
    cin>>a>>b;
    cout<<(a+b);
}
```

Note:

Iostream file : This directive causes the pre-processor to add the contents of iostream to the program . it contains declaration for identifiers. Cout and operator << and cin and operator >>.

- **Using >> or << more than one time in a statement is known as CASCADING.**
- **Example: cout<<a<<b; or cin>>a>>b;**

- **<< is called as insertion operator, which is used to insert values on the console(output screen)**

- **>> is called as extraction operator, which is used to extract values from key board**

- **DATA TYPES, VARIABLES, KEYWORDS, CONTROL STRUCTURES, OPERATORS etc.,, which are used in C are also applicable in CPP also.**

- **Apart from the 32 keywords in C we have some more keywords in CPP.**

<u>INLINE FUNCTION:</u>

An inline is a function i.e, expanded in line when it is invoked (called) i.e., the compiler replaces the function call with the corresponding function code.
Syntax :

inline datatyope function name(arg list)
{
Block of statements;
}

Example:

```
#include<iostream>
using namespace std;

inline square(int h)
{
    return h*h;
}

main()
{
    cout<< square(5);
}

output: 25
```

- In the above code when function call is made that is when square(5) is executed by compiler the function definition will be replaces the function call.

- This happens when a function is preceded by the keyword “inline”
- **Advantage** of inline function is that control of the program will be with main() only.
- **Disadvantage** is for each function call a separate copy of function definition is created in memory.

Note:

- Inline function may not work if it contains any loop (s),switch,goto,static varibles.
- Inline function cant be recurive.
- member function defined inside the class are inline

REFERENCE VARIABLE

- **A variable which is used to provide an alternative name for a previously defined variable is called as REFERENCE VARIABLE**.
- reference is a substitute for an object.
- (ampersand) & operator is used before the name of the variable.

Syntax : **datatype &referencevar_name=var_name;**

Example:

```
int x=10;
int &y=x;
cout<<x; //10
cout<<y; //10
```

REFERENCE TO A REFERNCE:

```
int x=10;
int &y=x;
int &m=y;
int &k=m;
cout<<x; //10
cout<<y; //10
cout<<m; //10
cout<<k  //10
```

it is not possible to assign a different value for a reference variable

Achieving Call by reference through reference parameter (variable):

```
Void function( int  &);
main()
{
int a=20;
function(a);
cout<<a;
}
void function ( int &b)  // b is pointing to the same location where a points
{
b=b+10;                                         output:
}                                                         30
```

UNARY SCOPE RESOLUTION OPERATOR (::)

- If the name of local and global variables is same then to differentiate both of them we use unary scope resolution operator.
- Scope resolution operator is denoted by **::**

Using unary scope resoulation operator, we can able to access the global variables when they have been hidden by the local variables of the same name in local scope.

Syntax :   :: VariableName ;

Example:

```
#include<iostream>
using namespace std;
int m=10; //global variable
main()
{
    int m=20;
```

```
        cout<<m;  //20
        cout<< ::m; //10
    }
```

- In the above ‘m’ is decalred as both local and global variable.
- ‘m’ refers to local value that is 20.
- ‘::m’ refers to global value that is 10.